CAPÍTULO 10

CLASIFICACIÓN DE IMÁGENES CON BAG OF VISUAL WORDS

Víctor GONZÁLEZ-CASTRO[1], Enrique ALEGRE[2], Eduardo FIDALGO[2]

[1] Department of Neuroimaging Sciences, Centre for Clinical Brain Sciences, University of Edinburgh, Edinburgh, United Kingdom
[2] Departamento de Ingeniería Eléctrica y de Sistemas y Automática, Universidad de León, León, España

La clasificación de imágenes es un proceso mediante el cual un ordenador es capaz de decidir qué contenidos están presentes en una imagen, esto es a qué clase pertenece o qué objetos contiene. En los últimos años el modelo Bag of Visual Words (BoVW) se ha convertido en una de las soluciones más utilizadas para realizar esta tarea. El término *visual word* (palabra visual, o simplemente "palabra") hace referencia a una pequeña parte de una imagen. El BoVW consta de varias etapas: un muestreo de puntos característicos (*keypoints*) de la imagen, la descripción de los mismos, la creación de un diccionario de palabras visuales mediante un proceso de agrupamiento, la representación de las imágenes a nivel global utilizando este diccionario y, finalmente, una clasificación de estas representaciones para decidir la clase a la que pertenece. En este capítulo se explicará el modelo BoVW de clasificación de imágenes, detallando estas etapas.

10.1. Introducción

La clasificación de imágenes consiste en determinar automáticamente los contenidos que aparecen en una imagen o, dicho de otro modo, asignar una o varias etiquetas a una imagen de entrada de entre un conjunto fijo de categorías. Este es uno de los problemas principales en Visión por Computador y, de hecho, muchas tareas de esta disciplina (segmentación, detección de objetos, detección de rostros) se pueden reducir a un problema de clasificación de imágenes. Por ejemplo, supóngase un sistema de clasificación de imágenes en 4 clases, *{gato, perro, cobaya, hamster}*, figura 10.1. Hay que tener en cuenta que, para un ordenador, una imagen se representa como una matriz numérica de tres

dimensiones (en el caso de imágenes en color) o una (en caso de imágenes en escala de grises), con valores que representan la intensidad, mayor o menor, de color o nivel de gris. La tarea del sistema de clasificación es obtener a partir de este gran conjunto de datos numéricos una etiqueta de categoría.

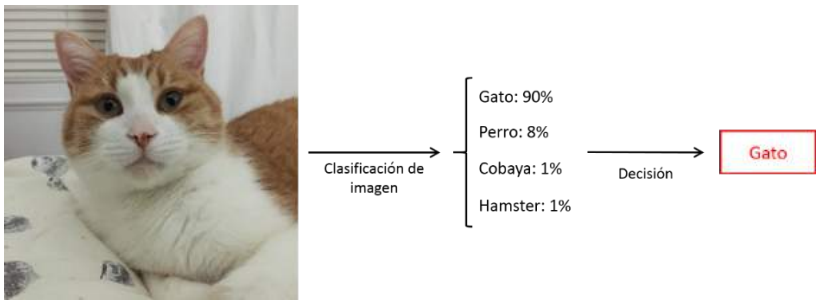
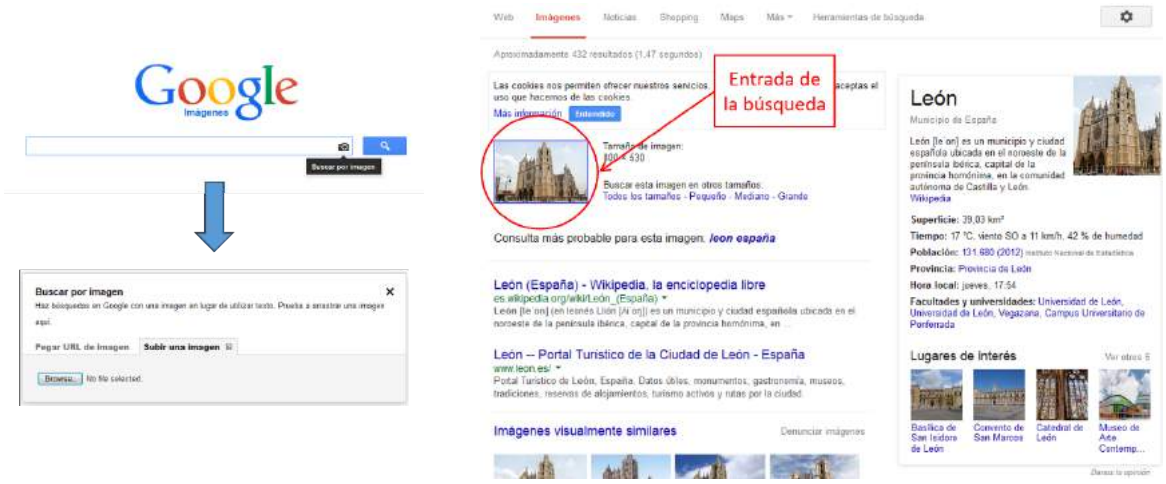


Figura 10.1. Ejemplo de sistema de clasificación de imágenes en las categorías *{gato, perro, cobaya, hamster}*.

La clasificación de imágenes se encuentra presente en muchas aplicaciones que utilizamos en la vida diaria. Facebook ha implementado detectores de personas y rostros para sus aplicaciones, y Google ha desarrollado clasificadores de imágenes más sofisticados para sus Google Glasses, o para su buscador, cuando se selecciona la opción de búsqueda por imagen, figura 10.2. La clasificación de imágenes también está presente en otras aplicaciones en la industria como son la teledetección (Sheng y col., 2010), la seguridad (Jing y col., 2013), la clasificación de documentos (Augereau y col., 2014), etc.



(a) *(b)*

Figura 10.2. Aplicación "buscar por imagen" de google imágenes (*a*) Interfaz para subir una imagen. (*b*) Resultados de la búsqueda.

A pesar de que reconocer una imagen es trivial para un ser humano, hay que considerar los retos que presenta para un sistema automático de Visión por Computador para el que, recordemos, una imagen no es más que un conjunto de números. Esa dificultad se basa principalmente en la presencia de uno o varios de los siguientes aspectos:

- **Cambio en el punto de vista:** Un mismo objeto puede estar orientado de muchas maneras diferentes respecto a la cámara.
- **Variación en la escala:** Los objetos de una misma clase a menudo presentan variaciones en su tamaño (por ejemplo, la altura de una persona puede ser muy variable).
- **Cambios en la pose:** Muchos objetos no son rígidos, y puede aparecer en posiciones diferentes.
- **Oclusiones:** El objeto de interés puede estar parcialmente oculto tras otro objeto en una imagen.
- **Condiciones en la iluminación:** Los diferencias de la iluminación sobre una imagen pueden ocasionar grandes cambios de intensidad a nivel de píxel.
- **Variación intra-clase:** Las clases *objetivo* pueden contener objetos diferentes entre sí. Por ejemplo, la clase *silla* puede contener sillas muy diversas,
- **Similaridad inter-clase:** Clases *objetivo* diferentes pueden tener ciertos parecidos entre sí que "confundan" al sistema de clasificación. Por ejemplo, un *lobo* puede ser similar a ciertos *perros*.

10.1.1. Modelo Bag of Visual Words

Bag of Visual Words (BoVW) (Szelinski, 2011), basado en el modelo Bag of Words utilizado en procesamiento del lenguaje natural, es un modelo de clasificación de imágenes propuesto inicialmente por Sivic y Zisserman (Sivic y Zisserman, 2003), que representa una imagen en función de la frecuencia de una serie de elementos visuales. Se esperaría que, por ejemplo, una imagen de un edificio contenga más ventanas que una imagen de un árbol. Estos elementos visuales se llamarán palabras visuales. El conjunto de palabras visuales constituye el diccionario.

Un sistema de clasificación de imágenes que utilice BoVW contiene los siguientes elementos:

1. **Conjunto de imágenes o *dataset***. Para crear un modelo de BoVW es necesario disponer de conjuntos de imágenes, etiquetados y normalmente grandes, también llamados *datasets*. Están formados por cientos, miles o incluso millones de imágenes de las que normalmente se conoce su contenido. Un ejemplo de *dataset* ampliamente utilizado en la investigación es *Flowers* (Nilsback y Zisserman, 2008), formado por 1360 imágenes que contienen 17 tipos de flores, estando cada tipo representado por 80 imágenes.

2. **Muestreo de puntos característicos o *keypoints*:** En cada imagen del *dataset*, es necesario seleccionar una serie de características que permitan representarla. Para ello es necesario seleccionar las posiciones en las que se extraerán dichas características. Esos puntos clave se suelen obtener principalmente mediante el cálculo de puntos característicos (*keypoints*), un muestreo denso o un muestreo aleatorio.

3. **Descripción de regiones alrededor de puntos característicos:** Los puntos clave seleccionados se caracterizan mediante algún descriptor que típicamente recoge la información de cómo se distribuyen los niveles de grises de los píxeles que le rodean. Algunos ejemplos de descriptores típicamente utilizados son SIFT (Lowe, 2004), SURF (Bay y otros, 2008), HoG (Dalal y Triggs, 2005), etc.

4. **Creación de un conjunto de entrenamiento y pruebas:** Es necesario dividir el *dataset* en dos conjuntos disjuntos, uno de entrenamiento y otro de pruebas o *tests*. Habitualmente el conjunto de entrenamiento suele ser mayor al de test, con objeto de dar al sistema más robustez, pero no es imprescindible. Para el anterior *dataset* (Flowers) es posible hacer un conjunto de entrenamiento con el 75% de las imágenes (1020) y un conjunto de pruebas con el 25% restante (340). Ninguna imagen del conjunto de pruebas ha de estar incluida en el conjunto de entrenamiento y viceversa.

5. **Formación del diccionario:** Los descriptores extraídos en el punto 3 se usan para construir un diccionario, o conjunto de palabras que representen a los puntos clave del punto 2. Estos conjuntos de palabras representativas se obtienen agrupando los vectores de características obtenidos para cada punto clave mediante alguna técnica de *clustering* (K Means, Gaussian Mixture Model, etc.). Para la construcción del diccionario se usan única y exclusivamente los descriptores pertenecientes a las imágenes del conjunto de entrenamiento.

6. **Representación de imágenes:** Una vez que el diccionario está construido, cada imagen del *dataset* se representa mediante un histograma que contiene con qué frecuencia aparece en esa imagen cada una de las palabras visuales del diccionario. Para ello se mide la distancia de cada descriptor extraído de una imagen con todas las palabras visuales del diccionario: dicho descriptor quedará representado por la palabra visual cuya distancia sea menor. Repetido este proceso para todos los descriptores, cada imagen del *dataset* quedará representada por un vector cuyo tamaño será el mismo que el número de palabras visuales del diccionario. Cada elemento del vector indicará la frecuencia de esa palabra dentro de la imagen correspondiente.

7. **Aprendizaje y reconocimiento:** Usando las imágenes del conjunto de entrenamiento, se creará un modelo usando las imágenes codificadas con la frecuencia de cada palabra visual. La precisión de este clasificador se evalúa típicamente utilizando las imágenes del conjunto de pruebas.

En las figuras siguientes se ilustra este muestreo y formación del diccionario (figura 10.3), representación (figura 10.4) y el reconocimiento de imágenes (figura 10.5), respectivamente.

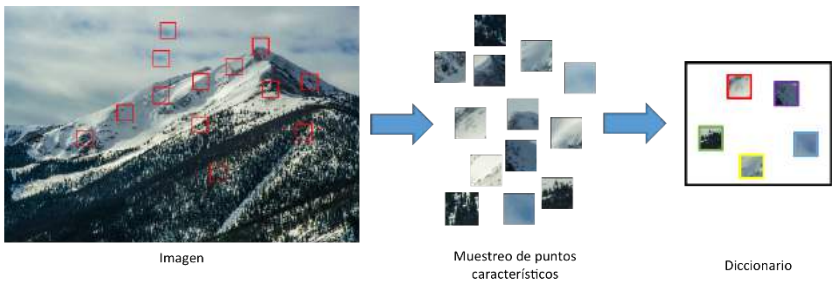


Figura 10.3. Ilustración del muestreo de características y formación del diccionario.

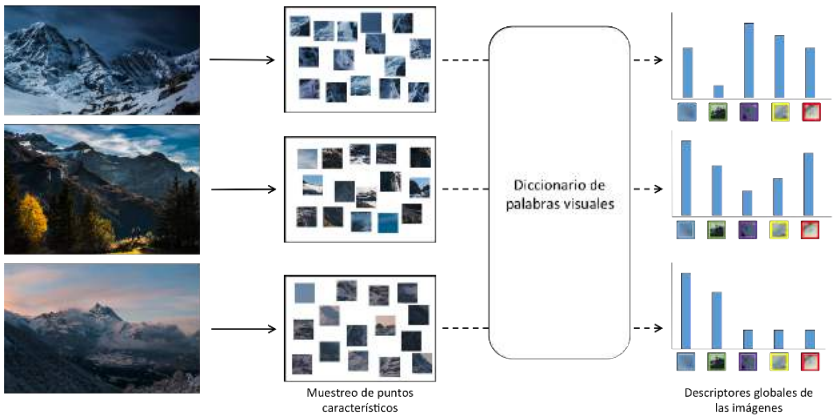


Figura 10.4. Diagrama de la etapa de representación de imágenes.

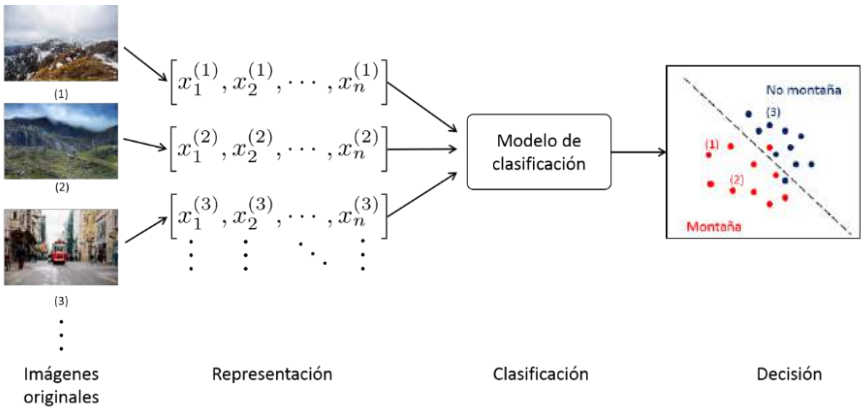


Figura 10.5. Diagrama de la etapa de reconocimiento de imágenes.

10.2. Descripción de imágenes utilizando SIFT

SIFT (Scale Invariant Feature Transform) es un método propuesto por Lowe (Lowe, 1999; Lowe, 2004) que permite obtener puntos característicos de la imagen y que posteriormente describe utilizando

un conjunto de histogramas de gradiente locales obtenidos a partir de la región que rodea a cada dicho punto característico. Los métodos que como SIFT utilizan este enfoque, obtener puntos invariantes y describir la región que los rodea, son muy robustos ante solapamientos parciales del objeto, variaciones de escala, translaciones y cambios en la iluminación. Por ese motivo estos métodos han sido muy utilizados en los últimos años en un gran número de aplicaciones que requieren describir y reconocer objetos o incluso clasificar imágenes, como es el problema que nos ocupa en este capítulo. Si bien, desde que SIFT fue publicado por primera vez (Lowe, 1999) han surgido muchos otros métodos similares que utilizan la misma estrategia basada en obtener características locales invariantes de la imagen, SIFT sigue siendo a día de hoy el referente y también, muy probablemente, el más utilizado.

El método propuesto por Lowe (Lowe, 1999) consta de dos partes claramente diferenciadas: la obtención de *puntos característicos* y la *descripción* de la región alrededor de cada punto de interés. Como se ha comentado anteriormente los dos primeros pasos en la creación de un modelo basado en Bag of Visual Words son el muestreo de puntos característicos y la descripción de cada uno de estos puntos. Si bien la obtención de la posición de los puntos puede realizarse de diferentes formas, en el contexto de la clasificación de imágenes, se asume que el *muestreo denso* es el procedimiento a seguir. Por ello, en esta sección explicaremos el método propuesto por Lowe (Lowe, 1999) para la obtención de puntos característicos, posteriormente indicaremos cómo se realiza un muestreo denso y finalmente explicaremos cómo se obtiene un descriptor para una ubicación determinada, siguiendo el método propuesto por Lowe (Lowe, 1999; Lowe, 2004).

10.2.1. Obtención de puntos característicos

Uno de los motivos por los que SIFT es muy utilizado es porque llevó un paso más allá la detección y localización de puntos característicos en el espacio y en la escala. Si bien el concepto de pirámides de paso banda de diferencia de Gaussianas había sido propuesto por Burt (Burt y Adelson, 1983) y por Crowley (Crowley y Stern, 1984), Harris (Harris y Stephens, 1988) ya había realizado su propuesta para detectar esquinas en base a las relaciones entre la traza y el determinante del Hessiano, y Lindeberg (Lindeberg, 1994) ya había demostrado que el único kernel posible para el espacio-escala es la función gaussiana, Lowe combinó todas estas ideas para proponer una nueva forma de obtener puntos característicos invariantes a la escala.

Los pasos necesarios a seguir para calcular dichos puntos son los siguientes:

1. **Detección de extremos** en el espacio-escala

   Se buscan puntos de interés en toda la imagen y en todas las escalas consideradas utilizando un espacio-escala formado por diferencias de Gaussianas.

2. **Localización precisa** de puntos característicos

Se localizan de forma más precisa los extremos anteriores, ajustando al punto obtenido una función cuadrática 3D. Posteriormente, se eliminan *keypoints* que están próximos a los bordes o tienen bajo contraste.

3. Asignación de la **orientación**

   A cada punto característico se le asigna una o varias *orientaciones* en función de las *direcciones del gradiente local*. Esta orientación, conjuntamente con la ubicación y la escala calculadas anteriormente, permiten que el descriptor sea invariante a estas tres transformaciones.

10.2.1.1 El espacio escala

Para detectar puntos característicos tanto en el espacio como en la escala, Lowe propuso crear un espacio escala mediante diferencia de Gaussianas. En contraste con otros métodos como el de Lindeberg (Lindeberg, 1994) que utilizaba el Laplaciano de la Gaussiana normalizado a la escala, o el mismo Harris (Harris y Stephens, 1988), que utilizaba un criterio basado en la relación entre el determinante y la traza del Hessiano, Lowe propone utilizar DoG, diferencia de Gaussianas, para ubicar puntos característicos, tanto en el espacio como en la escala. Aunque DoG no sea el mejor detector desde el punto de vista de la repetibilidad, como probó Mikolajczyk (Mikolajczyk y Schmid, 2002) sí es muy eficiente, más que el LoG, es una buena aproximación al Laplaciano de Gaussiana normalizado a la escala y es muy estable en los puntos detectados.

Por ello, Lowe (Lowe, 1999) propuso crear el espacio escala para SIFT mediante 4 octavas y 5 escalas cada octava con las siguientes características:

- Cada octava está formada por 5 imágenes, cada una procedente de una Gaussiana con una σ mayor que la anterior al ser multiplicada por un factor k. Se dice que cada una de esas imágenes se encuentra a una escala diferente, determinada por el valor de su σ.
- Cada octava se diferencia de la anterior en el tamaño de la imagen ya que la segunda octava se obtiene reduciendo el tamaño de las imágenes en la octava anterior por un factor de dos. En concreto, se obtiene eliminando una de cada dos filas y columnas. De igual manera, para obtener la tercera octava se realiza el mismo procedimiento para reducir el tamaño de las imágenes de la segunda. Y así sucesivamente.

10.2.1.2 Localización precisa de los puntos de interés

En el artículo publicado por Lowe (Lowe, 2004) se añadió al método presentado originalmente cinco años antes una propuesta que permite ajustar una función cuadrática 3D a los puntos de una muestra local, de manera que se puede determinar la localización interpolada del máximo. Esta

propuesta, basada en el método publicado por Brown (Brown y Lowe, 2002) proporciona a SIFT una mejora en la búsqueda de correspondencias y en la estabilidad al eliminar puntos de bajo contraste.

En ocasiones alguno de los puntos extremos detectados cae a lo largo de un borde lo que hace que sean puntos poco deseados al ser menos estables. Para eliminarlos, SIFT utiliza la relación entre los valores propios del Hessiano y, para evitar calcular explícitamente sus valores, toma prestada de Harris (Harris y Stephens, 1988) la aproximación que permite obtener la relación entre dichos valores propios a partir del cálculo de la traza y del determinante del Hessiano. De esa manera, Lowe propone evaluar la siguiente expresión:

$$\frac{Tr(H)^2}{Det(H)} < \frac{(r+1)^2}{r}, \tag{10.1}$$

donde r es la relación entre los dos valores propios y que en SIFT se fija como $r = 10$. Aquellos puntos clave con una relación entre las curvaturas principales mayor de 10, son eliminados.

10.2.1.3 Asignación de la orientación

Para conseguir que el descriptor, que se calculará posteriormente alrededor del punto característico, sea invariante a la rotación, es necesario en primer lugar asignar una orientación principal a cada *keypoint* y después tenerla en cuenta en el cálculo del descriptor.

Con el fin de que el cálculo de la orientación sea invariante a la escala, se selecciona la imagen Gaussiana suavizada que tiene la escala más próxima a la del punto característico que se está estudiando. Para cada píxel de dicha imagen, se calcula su magnitud, $m(x, y)$, y orientación, $\theta(x, y)$, utilizando diferencias entre los píxeles vecinos, de la siguiente manera:

$$\begin{aligned} m(x,y) &= \sqrt{(L(x+1,y)-L(x-1,y))^2 + ((L(x,y+1)-(L(x,y-1))^2}, \\ \theta(x,y) &= \tan^{-1}\frac{L(x,y+1)-L(x,y-1))}{L(x+1,y)-L(x-1,y))} \end{aligned} \tag{10.2}$$

Tomando la región que rodea al punto característico, se calcula un histograma de 36 elementos, donde cada elemento cubre un sector circular, consecutivo al anterior, de 10º. En cada una de las 36 agrupaciones del histograma se añade la magnitud del gradiente correspondiente a esa orientación previamente ponderado por una gaussiana circular, centrada en el punto característico, y cuya σ es 1,5 veces el tamaño de la escala a la que se localizó dicho punto.

Un detalle adicional del método es que a cada punto de interés se le asignan una o varias orientaciones, de manera que aproximadamente un 15% de los puntos obtenidos tendrán múltiples orientaciones. Estos puntos, con más de una orientación asignada, contribuyen muy positivamente a que la búsqueda de correspondencias sea estable. Para realizar esta asignación, sobre el histograma anterior se detecta el pico más alto y todos aquellos que tengan una magnitud superior al 80% de dicho pico. Se crea un punto característico para cada uno de los picos anteriores, con la orientación indicada por su posición en el histograma.

El último paso consiste en ajustar una parábola a los 3 valores del histograma que están más próximos a cada pico elegido de manera que se interpola la posición del pico de una forma más precisa.

10.2.2. Muestreo denso de puntos de interés

Como se ha comentado al principio de esta sección, el muestreo denso es el método normalmente empleado cuando los descriptores obtenidos con SIFT se utilizan para clasificar imágenes. Este muestreo se caracteriza por que no se buscan puntos de interés sino que se calcula un descriptor SIFT, en una ventana de tamaño *s*, para cada *n* píxeles de la imagen. Si se siguiera la propuesta de Lowe para SIFT, el tamaño de la ventana, *s*, sería un entorno de 16x16 píxeles alrededor del punto de interés.

La idea subyacente en el muestreo denso es calcular un descriptor SIFT para cada *n* píxeles de la imagen donde, si *n* fuera igual a 1, se estaría calculando dicho descriptor para cada uno de los píxeles de la imagen. Dependiendo de la aplicación, este *n*, normalmente llamado el *step*, suele tener valores que van de 3 a 7, de forma que se calcularía el descriptor SIFT cada 3 o cada 7 píxeles de la imagen. Los valores del *step* suelen mantenerse bajos.

Los detalles de cómo se realiza este muestreo denso dependen de la implementación que se utilice. La librería VLFeat, (Vedaldi y Fulkerson, 2008) ofrece una función, vl_dsift que permite realizar este cálculo y de la que comentamos a continuación alguna de sus características, para ilustrar mejor alguna de las posibilidades que suelen considerarse cuando se muestrea una imagen de esta manera.

En la implementación de VLFeat, para obtener el muestreo denso, no se calcula un espacio escala gaussiano de la imagen. Esto es así porque se asume que dicho cálculo es necesario para la detección de puntos de interés y el muestreo denso se basa en elegir un gran número de puntos igualmente espaciados, que cubran toda la imagen, y que permitan describir su contenido. De todas formas, asumiendo que este muestreo denso puede realizarse a diferentes escalas, vl_dsift sugiere que se suavice previamente la imagen sobre la que se va a realizar el muestreo, al nivel deseado de escala, utilizando otra de las funciones de la librería.

La función vl_dsift llama *step* a la distancia en píxeles entre cada descriptor extraído, siendo su valor por defecto 1, y *size* al tamaño de la ventana sobre la que se calcula el descriptor. Otras opciones de esta función es que permite indicar la región sobre la que se calcula el muestreo mediante la opción *bounds,* o que se puede elegir entre utilizar una función Gaussiana sobre la ventana de cálculo, al igual que SIFT, o hacerlo de forma rápida, mediante la opción *fast,* y realizar el cálculo de forma plana, elemento a elemento, sin realizar la ponderación.

Como apunte final, indicar que cuando se utiliza muestreo denso, en contraste con lo que se hace en SIFT basado en puntos característicos, suele recomendarse suavizar poco la imagen original, para potenciar que los gradientes aparezcan resaltados.

10.2.3. Cálculo del descriptor SIFT

Una vez finalizado el proceso de detección de puntos característicos, o bien seleccionadas las posiciones de la imagen alrededor de las cuales se desea obtener un descriptor, se procede al cálculo del vector de características, típicamente de 128 elementos, basado en la concatenación de histogramas orientados de gradientes. Los puntos de interés son en realidad una tupla de cuatro elementos, *(x, y, σ, θ)*, donde (x,y) corresponde con las coordenadas del punto, σ con la escala a la que se ha obtenido y θ la orientación principal que tiene dicho punto, calculada según se explicó en la sección 10.2.1.3 Asignación de la orientación.

De forma resumida el cálculo del descriptor, para cada punto, consta de los siguientes pasos:

4. Se obtiene la *magnitud y la orientación del gradiente* en una ventana de, típicamente, 16x16 píxeles centrada en el punto característico.

5. Se calcula un *histograma de orientaciones del gradiente de 8 bins para cada una de las 16 regiones* que se obtienen al dividir la ventana inicial en regiones menores de 4x4 píxeles. Cada *bin* corresponde con uno de los 8 sectores circulares de 45 grados en los que se divide la circunferencia completa. El histograma contendrá las magnitudes acumuladas de los píxeles cuya orientación se encuentre dentro del sector asignado a cada *bin*.

6. Se *normaliza el vector de 128 elementos* que se obtiene al concatenar los histogramas de 8 valores para cada uno de las 16 divisiones que se han realizado de la región. Se realiza una doble normalización a longitud unidad.

A continuación se explican algunos detalles adicionales de cada uno de estos pasos.

10.2.3.1 Obtención de la magnitud y orientación del gradiente

Dada una ventana de 16 x 16 píxeles, alrededor del punto característico se obtiene la orientación del gradiente, para cada píxel, teniendo en cuenta los siguientes detalles:

- Se obtiene la magnitud y la orientación del gradiente para cada píxel, utilizando la información de sus vecinos horizontal y vertical. Se sigue el mismo procedimiento de cálculo que se realizó para obtener los gradientes en la etapa de cálculo de la orientación del punto característico según se explicó en la sección 10.2.1.3 Asignación de la orientación.
- El cálculo se realiza sobre la imagen del espacio escala con el nivel de suavizado Gaussiano más próximo a la escala del punto.

- Se realiza con invarianza a la orientación, rotando las coordenadas del descriptor y las orientaciones del gradiente en función de la orientación del punto característico.
- Se ponderan todos los valores de la magnitud del gradiente de la ventana a calcular por una gaussiana con una σ igual a la mitad del ancho de la ventana del descriptor.

10.2.3.2 Histograma de orientaciones del gradiente

Una vez se dispone de la orientación y la magnitud del gradiente para cada píxel se procede a calcular un histograma de las orientaciones de dicho gradiente en subregiones de 4x4 píxeles. Asumiendo que la ventana que se quiere describir tiene 16 x 16 píxeles, se divide ésta en 16 regiones menores, de 4 x 4 píxeles cada una. Para cada una de esas regiones se hace lo siguiente:

3. Se obtiene un *histograma de 8 direcciones para cada subregión*. Para ello se dividen los 360 grados de la circunferencia en 8 sectores circulares de 45 grados cada uno. Las orientaciones que caigan en el primer sector, de 0 a 45 grados, corresponderán con el primer *bin* del histograma, y así sucesivamente. Cada *bin* del histograma tendrá un valor correspondiente a la suma de las magnitudes de todos los píxeles cuya orientación corresponda con su sector circular.

4. Se *distribuye el valor de cada gradiente en los bins adyacentes* del histograma, mediante una interpolación tri-lineal, de forma que se evitan los efectos de frontera. Finalmente, cada una de las entradas a un *bin* del histograma se multiplica por un peso de $1 - d$ para cada dimensión, donde d es la distancia de la muestra al valor central del *bin* medida en unidades del espaciado de los *bins* del histograma.

Una vez obtenido el histograma orientado de gradientes para cada una de las 16 subregiones en las que se había dividido la ventana, se concatenan todos los histogramas formando un vector de características de 16 x 8 = 128 elementos.

10.2.3.3 Normalización del vector de características

El paso final consiste en normalizar el vector obtenido para mejorar la robustez del anterior descriptor frente a ciertos cambios de iluminación, producidos por la reflexión de la luz en superficies 3D o por efectos debidos a la saturación de la cámara.

SIFT realiza una doble normalización a longitud unidad de la siguiente manera:

- En primer lugar se realiza una primera normalización a longitud, dividiendo el vector por su norma.
- Después se umbraliza cada uno de los elementos del vector anterior, utilizando 0.2 como valor de corte, de manera que todos aquellos elementos del vector que tengan un valor

superior al de corte, se dejan con 0.2. Finalmente se vuelve a realizar una segunda normalización a longitud unidad, obteniendo de esta manera el vector definitivo.

Con todo lo anterior, el vector normalizado de 128 elementos conteniendo los histogramas de las orientaciones de los gradientes, queda calculado.

10.3. Generación del diccionario

La generación del diccionario consiste en obtener, a partir de N vectores de características (descriptores) extraídos de un conjunto de imágenes de entrenamiento, K palabras visuales mediante alguna técnica de agrupamiento (*clustering*) de datos, figura 10.6. Es importante resaltar que el diccionario no se puede generar usando descriptores de imágenes del grupo de pruebas.

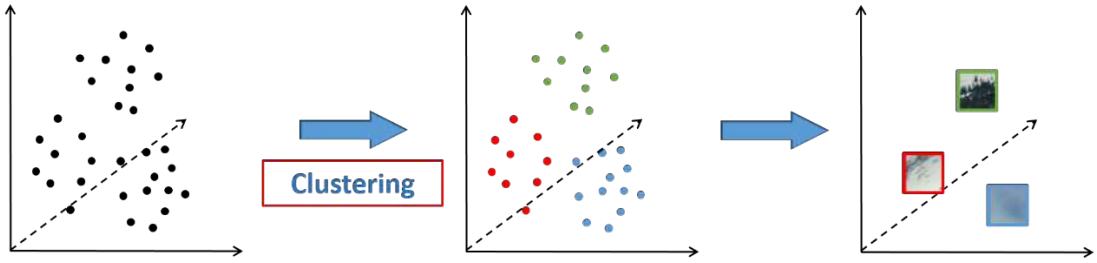


Figura 10.6. Diagrama de obtención de $K = 3$ palabras a partir de un conjunto de descriptores.

Los algoritmos de *clustering* agrupan un conjunto de elementos, representados por vectores de n características de acuerdo con un criterio, generalmente de distancia o similitud en el espacio de características. Idealmente, los agrupamientos deben ser tales que: (a) sean compactos (sus elementos son cercanos en el espacio de características) y (b) los distintos grupos estén lo más lejos posible entre ellos.

Existen varios tipos de algoritmos de *clustering*. Algunos de los más típicos son:

- ***Clustering* basado en centroides:** En estos algoritmos, cada agrupamiento está representado por un vector de características único, que no tiene que pertenecer al conjunto de datos original. Algunos ejemplos de este tipo de *clustering* son k-means, k-medians, k-medoids, fuzzy k-means, etc.

- ***Clustering* jerárquico:** Cuanto más relacionados entre sí están dos elementos, más cercanos se encuentran en el espacio de características. Siguiendo esa idea, este tipo de algoritmos "conectan" elementos para formar agrupamientos basados en sus distancias, de forma que un grupo se puede describir mediante la distancia necesaria para conectar sus elementos más lejanos. A menudo los agrupamientos obtenidos con este tipo de algoritmos se representan mediante dendrogramas. Un nuevo elemento se añade al grupo utilizando un criterio de unión o *linkage* que contempla la distancia del elemento a añadir al centroide del grupo, a su elemento más cercano, más alejado, etc. En función de ello se habla de diferentes métodos

de unión, o *linkage*, como son *single-linkage-clustering*, *complete linkage clustering*, UPGMA, etc.

- ***Clustering* basado en distribuciones:** Los algoritmos de agrupamiento de este tipo se basan en la consideración de que los elementos de un mismo grupo pertenecen a la misma distribución de probabilidad. Un método muy popular dentro de este tipo de algoritmos es el Gaussian Mixture Model (usando el algoritmo Expectation-Maximization, o EM), que trata de agrupar los datos asumiendo diferentes distribuciones normales, y variando los parámetros iterativamente hasta conseguir la convergencia.

- ***Clustering* basado en densidad:** En este tipo de algoritmos los grupos se definen como áreas de densidad más alta (en el espacio de características) que el resto de datos. Los datos que no pertenecen a estas regiones con alta densidad se consideran ruido o puntos "frontera" entre distintos grupos. Ejemplos de algoritmos de *clustering* basado en densidad son DBSCAN, OPTICS, o Mean-Shift.

En este capítulo sólo se explicará en detalle el algoritmo k-means, propuesto originalmente en (Hartigan, 1979). La razón es que este algoritmo representa cada agrupamiento de descriptores (palabra visual) mediante un vector de características (el prototipo, o centroide, del *cluster*), siendo este tipo de entrada la más adecuada para el proceso de clasificación posteriormente explicado.

10.3.1. Algoritmo de *clustering* K-means

Supongamos que tenemos un conjunto de datos $\{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_N}\}$ que consiste en N observaciones de una variable D-dimensional $\mathbf{x}$. El objetivo es dividir los datos en un número de *clusters* K, que vamos a suponer dado. Como ya se indicó anteriormente, se considera que los elementos de un *cluster* se sitúan a distancias pequeñas entre ellos en comparación con las distancias a los puntos que no pertenecen a ese *cluster*. Formalizaremos esta noción introduciendo un conjunto de vectores D-dimensionales $\boldsymbol{\mu}_\mathbf{k}$, con $k = 1, 2, \ldots, K$, de forma que $\boldsymbol{\mu}_\mathbf{i}$ es un prototipo asociado al *cluster* i-ésimo. El objetivo, por lo tanto, es calcular (a) los vectores $\{\boldsymbol{\mu}_\mathbf{k}\}$ y (b) una asignación de los datos a dichos vectores de manera que la suma de los cuadrados de las distancias entre cada punto $\mathbf{x_n}$ y el representante $\boldsymbol{\mu}_\mathbf{i}$ del *cluster* al que se ha asignado sea mínima. Esto se puede llevar a cabo mediante un proceso iterativo en el que en cada repetición se realicen dos pasos: Se comienza asignando unos valores iniciales a los vectores $\{\boldsymbol{\mu}_\mathbf{k}\}$. A continuación, en una primera fase, se asigna cada punto $\mathbf{x_n}$ al *cluster* cuyo prototipo $\boldsymbol{\mu}_\mathbf{i}$ sea más cercano. En una segunda fase se recalcula cada prototipo $\boldsymbol{\mu}_\mathbf{i}$ como la media de los puntos $\mathbf{x_n}$ asignados al *cluster* correspondiente.

Ambas fases se repiten hasta que las asignaciones no cambien, o bien hasta que se llegue a un número máximo de iteraciones. La convergencia de este algoritmo está garantizada debido a que en cada iteración se reduce la suma de las distancias entre los puntos y los representantes del *cluster* al

que pertenecen o, en el peor caso, esta suma no cambia. Sin embargo, el algoritmo puede converger a un mínimo local en vez de a un mínimo global.

La figura 10.7 ilustra las etapas del algoritmo K-means:

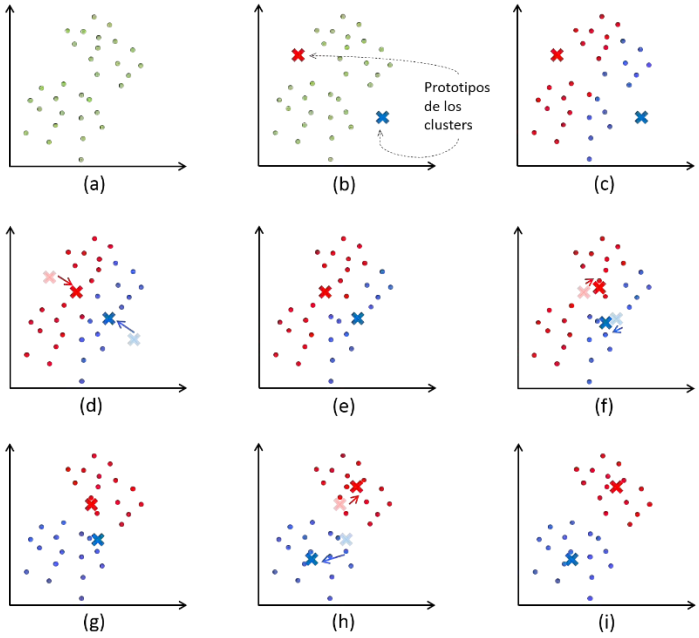


Figura 10.7. Ilustración del algoritmo K-means. (a) Los puntos verdes denotan los datos en un espacio Euclídeo bidimensional. (b) Inicialización de los prototipos μ_1 y μ_2 (cruces roja y azul). (c) Asignación inicial de cada punto a uno de los *clusters*. Cada punto se asigna al *cluster* rojo o azul en función de la cercanía de los prototipos correspondientes. (d) Actualización de los prototipos como la media de los nuevos puntos asignados a cada *cluster*. (e) - (i) Sucesivos pasos hasta la convergencia final del algoritmo.

Nótese que en el ejemplo anterior se han inicializado puntos no óptimos deliberadamente. En la práctica un método mejor de inicialización será escoger como prototipos $\{\boldsymbol{\mu}_k\}$ un subconjunto de K puntos de entre el conjunto de datos original. De hecho, la elección aleatoria de puntos podría dar lugar a que no se asignase ningún punto a un prototipo determinado, como se puede ver en la figura 10.8.

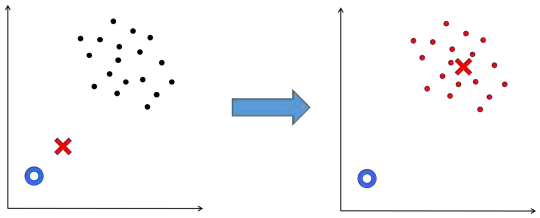


Figura 10.8. Ilustración de inicialización incorrecta de prototipos: ninguno de los puntos del conjunto de datos se asignará al *cluster* representado en rojo.

Esta es una breve introducción al método K-means. Una explicación más detallada puede encontrarse en la sección 9.4.1 del capítulo 9, “Clasificación y Reconocimiento de patrones”.

10.4. Representación de imágenes

Una vez creado el diccionario de palabras visuales mediante alguna técnica de *clustering* tal como K-means es el momento de representar la imagen. En el modelo Bag of Visual Words de clasificación de imágenes, representar una imagen consiste en describirla de manera global mediante un histograma de ocurrencias de palabras visuales en el diccionario. Dicho de otro modo, para cada descriptor de la imagen es necesario buscar la palabra del diccionario más similar. Una vez encontradas dichas palabras para todos los descriptores de una imagen, ésta se construye mediante el histograma de frecuencias de las mismas. Para determinar cuan parecidas son dos palabras se utilizará una medida de distancia entre sus descriptores (por ejemplo SIFT). Este proceso se ilustra en la figura 10.9.

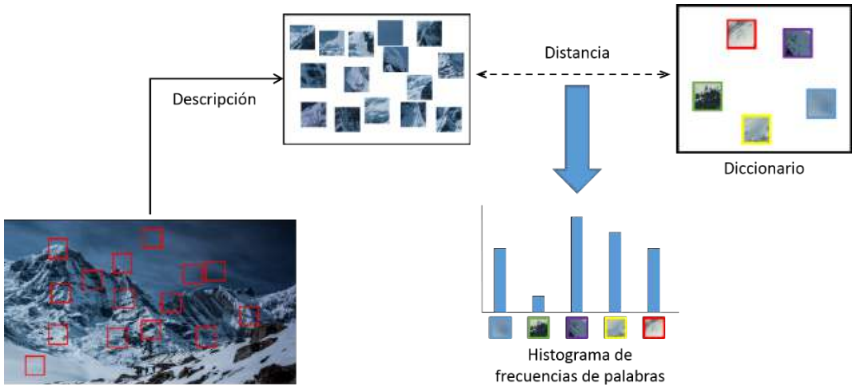


Figura 10.9. Ilustración del proceso de representación de imágenes.

En concreto, para cada uno de los n puntos característicos muestreados de la imagen se calculará la distancia entre su descriptor y los vectores que representan cada una de las K palabras del diccionario obtenido en el proceso de construcción del mismo (esto es el *clustering* de las palabras). Formalizando esta idea, sean $\mathbf{x}_i$ el vector que describe el i-ésimo descriptor de la imagen y μ_j el vector correspondiente a la j-ésima palabra del diccionario. Tras calcular la distancia entre ellos, $d_{ij} = \| \mathbf{x_i} - \boldsymbol{\mu}_\mathbf{j} \|_2$, $\forall j \in 1,\ldots K$ [1], se puede formar el vector p, con $p(i) = k$, donde k es el índice de la palabra $\boldsymbol{\mu}_\mathbf{k}$ que hace que $d_{ik} \leq d_{ij}$, $\forall j = 1,\ldots K$. Finalmente, el descriptor global (esto es la representación) de la imagen consiste en el histograma de frecuencias de p.

El resultado es un conjunto de vectores que contienen los histogramas de frecuencias de las K palabras visuales del diccionario. Cada vector está asociado a una imagen del *dataset*.

10.5. Clasificación

La última etapa del sistema es la de clasificación. Esta etapa consiste en decidir la clase a la que pertenece una imagen cualquiera a partir de sus descriptores. En el proceso de creación del modelo

[1] En este caso se muestra la distancia Euclídea, pero podría utilizarse cualquier otra medida de distancia (Manhattan, Mahalanobis, etc.)

sólo han participado las imágenes del conjunto de entrenamiento. Con ellas se ha creado el diccionario que ha servido para describir, a través del histograma de frecuencias de las palabras visuales, las imágenes del conjunto de prueba. Si se quiere evaluar el modelo creado, se realizará un proceso de test con las imágenes separadas previamente para ese fin. Esta evaluación da una idea de cómo se comportará este modelo a la hora de clasificar una imagen nueva, perteneciente a una clase desconocida. La evaluación, expresada típicamente mediante alguna medida basada en la precisión (precisión) o la rememoración (recall), como puede ser el F-Score, indica si el modelo creado generalizará bien. Como hemos indicado, cuando se utiliza Bag of Visual Words cada imagen se describe mediante su histograma de frecuencias de palabras visuales, siendo ese el vector de características utilizado.

En general, como se ha explicado en el capítulo 9, existen dos enfoques diferentes para crear un modelo:

1. **Aprendizaje no supervisado:** Los algoritmos de aprendizaje no supervisado tratan de ajustar un modelo de clasificación a partir de un conjunto de datos no etiquetados (de los que no se conocen las clases a las que pertenecen) $\{\mathbf{x_1},\ldots\mathbf{x_N}\}$. Para realizar este ajuste, el algoritmo debe encontrar de forma autónoma las características, regularidades y correlaciones de los datos, de forma que pueda separar las categorías similares entre sí. Algunos ejemplos de este tipo de aprendizaje son los algoritmos de *clustering* mencionados en la sección 10.3., *Self-Organising Maps* (SOM) o *Adaptive Resonance Theory* (ART).

2. **Aprendizaje supervisado:** Los algoritmos de aprendizaje supervisado ajustan el modelo, o función, de clasificación mediante un conjunto de datos cuya clase es conocida, llamado *conjunto de entrenamiento*. Formalmente este conjunto se representa como $\{(\mathbf{x_1},\mathbf{y_1}),\ldots(\mathbf{x_N},\mathbf{y_N})\}$, donde $\mathbf{x_i}$ es el *i*-ésimo vector del conjunto de entrenamiento mientras que $\mathbf{y_i}$ es la clase a la que pertenece dicho vector. Una vez ajustado el modelo, la función de clasificación es capaz de asignar la clase $\mathbf{y_k}$ a la que pertenecen nuevos elementos $\mathbf{x_k}$. El conjunto de elementos de clase desconocida se llama **conjunto de test**. Algunos ejemplos de algoritmos de aprendizaje supervisado son las redes neuronales artificiales, la regresión logística, los *k* vecinos más cercanos (k-Nearest Neighbours o kNN), las máquinas de vector de soporte (Support Vector Machines), etc.

Este capítulo explicará brevemente qué es y cómo se usa el aprendizaje supervisado mediante máquinas de vector de soporte (SVM) en la clasificación de imágenes.

10.5.1. Máquinas de Vectores Soporte (Support Vector Machines)

Las máquinas de vectores soporte (SVM por sus siglas en inglés *Support Vector Machines*) (Vapnik, 1995) constituyen una de las técnicas de clasificación supervisada más potentes, y un estándar en el estado del arte actualmente. Se trata de un clasificador biclase (esto es permite clasificar

en dos clases), aunque la mayoría de implementaciones del mismo, (como, por ejemplo, LIBSVM, LIBLINEAR) soportan clasificación multi-clase.

Al igual que cualquier clasificador supervisado biclase, el objetivo de un SVM es inferir una frontera de decisión (lineal o no) en el espacio de características de modo que las observaciones posteriores se clasifiquen automáticamente en uno de los dos grupos definidos por dicha frontera (también conocida como *hiperplano*). La particularidad de SVM es que trata de generar dicho hiperplano de modo que maximice su separación con cada uno de los grupos, figura 10.10. Por ello se dice que este clasificador es un clasificador de margen máximo.

Si consideramos elementos en un espacio D-dimensional, i.e. $\circ^{D}$, el hiperplano está determinado por un vector $\boldsymbol{\theta} = (\theta_0, \theta_1, ..., \theta_D)$, de forma que su ecuación es:

$$\theta_0 + \sum_{i=1}^{D} \theta_i X_i = 0 \tag{10.3}$$

Así, dado un elemento $\mathbf{x} = (x_1, \ldots, x_D) \in \circ^{D}$, un clasificador asigna $\mathbf{x}$ a una clase o a otra en función de si $\theta_0 + \sum_{i=1}^{D} \theta_i x_i > 0$ o $\theta_0 + \sum_{i=1}^{D} \theta_i x_i < 0$. Sin embargo, en el caso de SVM esta asignación se realiza en función de si $\theta_0 + \sum_{i=1}^{D} \theta_i x_i > \gamma$ o $\theta_0 + \sum_{i=1}^{D} \theta_i x_i < \gamma$ con $\gamma > 0$.

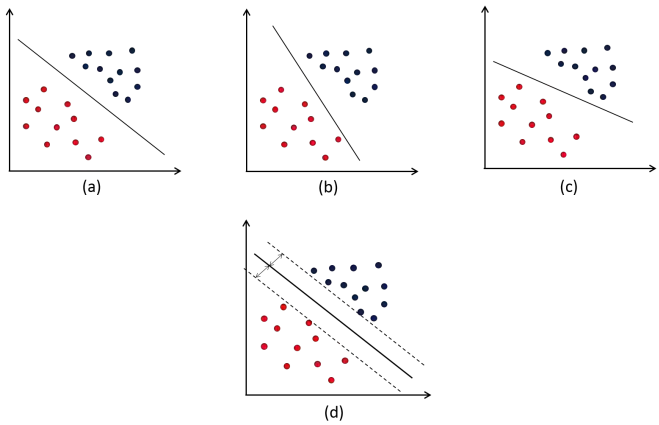


Figura 10.10. Ejemplos de hiperplanos que separan dos clases diferentes en un espacio bidimensional. (a)-(c) Diferentes hiperplanos. (d) Hiperplano de margen máximo.

Para inferir el hiperplano durante la fase de entrenamiento es necesario minimizar la función de coste:

$$J(\theta) = C \sum_{i=1}^{N} \left[\mathbf{y_i} cost_1(\boldsymbol{\theta}^T \mathbf{x_i}) + (1 - \mathbf{y_i}) cost_0(\boldsymbol{\theta}^T \mathbf{x_i}) \right] + \frac{1}{2} \sum_{j=1}^{D} \theta_j^2, \tag{10.4}$$

donde $cost_0(z)$ y $cost_1(z)$ son funciones de coste en función de si $\mathbf{y}_i = 1$ o $\mathbf{y}_i = 0$, respectivamente que tienen un aspecto similar al mostrado en la figura 10.11.

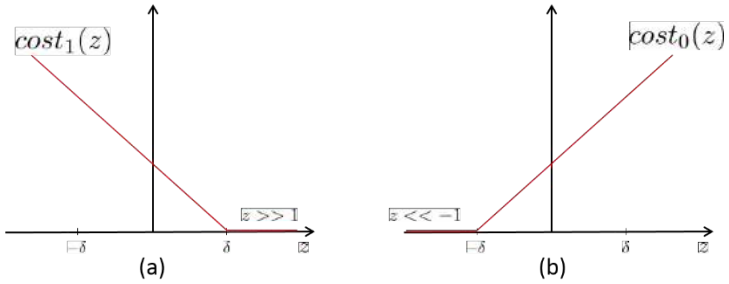


Figura 10.11. Ilustración de las funciones $cost_0$ y $cost_1$.

Otra de las particularidades de las SVM es su capacidad de obtener fronteras de decisión no lineales, figura 10.12 aumentando la dimensión del espacio de características mediante el uso de *kernels*. Por ejemplo, mapeando un conjunto de D características $\{X_1,\ldots,X_D\}$ en, por ejemplo, uno de $2D$ características $\{X_1,X_1^2,\ldots,X_D,X_D^2\}$. Esto es especialmente útil cuando las clases de los elementos que se van a clasificar **no son linealmente separables en el espacio original**. Este mapeado se lleva a cabo con las llamadas funciones *kernel* o, simplemente, kernels, $k(\mathbf{x},\mathbf{y})$, que se diseñan de forma que el valor que devuelven sea más pequeño cuanto más alejado esté $\mathbf{y}$ de $\mathbf{x}$. En la práctica, tanto los elementos $\mathbf{x}$ como $\mathbf{y}$ serán los elementos del conjunto de entrenamiento. En definitiva, los vectores que definen el hiperplano se calculan como una combinación lineal de imágenes del kernel k con parámetros α_i:

$$\sum_i \alpha_i k(\mathbf{x_i},\mathbf{x}) = cte \tag{10.5}$$

Algunos de los *kernels* más comunes son el lineal, ecuación (10.6), polinomial, ecuación (10.7), gaussiano, también conocido como *radial basis function*,ecuación (10.8) o sigmoidal, ecuación (10.9).

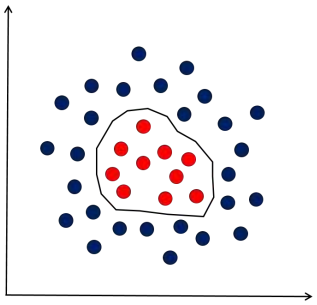


Figura 10.12. Ejemplo de frontera de decisión no lineal.

$$k(\mathbf{x_i},\mathbf{x_j}) = \mathbf{x_i}\cdot\mathbf{x_j} \tag{10.6}$$

$$k(\mathbf{x_i},\mathbf{x_j}) = (\mathbf{x_i}\cdot\mathbf{x_j})^d \tag{10.7}$$

$$k(\mathbf{x_i}, \mathbf{x_j}) = \exp(-\gamma \|\mathbf{x_i} - \mathbf{x_j}\|), \ \gamma > 0^2 \quad (10.8)$$

$$k(\mathbf{x_i}, \mathbf{x_j}) = \tanh(\kappa \mathbf{x_i} \cdot \mathbf{x_j} + c), \ \kappa > 0, \ c < 0 \quad (10.9)$$

Se remite al lector interesado en más detalles sobre SVM al capítulo 5 de (Pajares y de la Cruz, 2010).

10.6. Bibliografía

Augereau, O.; Journet, N.; Vialard, A.; Domenger, J.-P. (2014) Improving Classification of an Industrial Document Image Database by Combining Visual and Textual Features. Proceedings del 11th IAPR International Workshop on Document Analysis Systems (DAS), Tours, Francia, 7-10 Abril 2014; pp. 314–318.

Bay, H.; Ess, A.; Tuytelaars, T. and Van Gool, L. (2008) Speeded-Up Robust Features (SURF). Computer Vision and Image Understanding 110(3): 346-359. doi:10.1016/j.cviu.2007.09.014.

Bishop, C.M. (2006) *Pattern Recognition and Machine Learning*, 1ª ed.; Springer Verlag: Nueva York, USA, pp. 423–428.

Brown, Matthew and Lowe, David G. "Invariant features from interest point groups," British Machine Vision Conference, BMVC 2002, Cardiff, Wales (September 2002), pp. 656-665

Burt, Peter and Adelson, Ted (1983). The Laplacian pyramid as a compact image code. IEEE Transactions on Communications 9(4): 532–540. doi:10.1109/tcom.1983.1095851.

Crowley, James L. and Stern, Richard M. (1984). Fast computation of the difference of low pass transform. IEEE Transactions on Pattern Analysis and Machine Intelligence 6(2): 212-222. doi:10.1109/tpami.1984.4767504.

CS231n Convolutional Neural Networks for Visual Recognition: Image Classification, data-driven approach, k-nearest neighbor. Disponible online: http://cs231n.github.io/classification/

CS231n Convolutional Neural Networks for Visual Recognition: Linear classification: SVM/Softmax. Disponible online: http://cs231n.github.io/linear-classify/

Dalal, N.; Triggs, B. (2005) "Histograms of oriented gradients for human detection," Computer Vision and Pattern Recognition CVPR 2005. IEEE Computer Society Conference on , vol.1, no., pp.886,893 vol. 1, 25-25. doi: 10.1109/CVPR.2005.177

Duda, R.O.; Hart P.E.; Stork, D.G. (2000) *Pattern Classification*, 2ª ed.; John Wiley & sons: Canada, pp. 259–268.

Fei-Fei, L.; Fergus R.; Torralba A. Recognizing and Learning Object Categories, CVPR 2007 short course. Disponible online: http://people.csail.mit.edu/torralba/shortCourseRLOC/index.html

Harris, C. and Stephens, M. (1988). "A combined corner and edge detector". Proceedings of the 4th Alvey Vision Conference. pp. 147–151.

Hartigan, J.A.; Wong, M.A. (1979) Algorithm AS 136: A K-Means Clustering Algorithm. Journal of the Royal Statistical Society. Series C (Applied Statistics), 28(1), 100–108.

[2] En el kernel gaussiano en ocasiones se utiliza el valor $\gamma = 1/2\sigma^2$

Jing, Z.; Lei, S.; Li, Z.; Zhenwei, L.; Yuncong, Y. (2013) An approach of bag-of-words based on visual attention model for pornographic images recognition in compressed domain. *Neurocomputing*, 110, 145–152.

Lindeberg, T. 1994. Scale-space theory: A basic tool for analysing structures at different scales. Journal of Applied Statistics, 21(2):224-270.

Lowe, David G. (1999). "Object recognition from local scale-invariant features". Proceedings of the International Conference on Computer Vision 2. pp. 1150–1157. doi:10.1109/ICCV.1999.790410.

Lowe, D.G. (2004) Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision, 60(2), 91–110.

Mikolajczyk, K., and Schmid, C. 2002. An affine invariant interest point detector. In European Conference on Computer Vision (ECCV), Copenhagen, Denmark, pp. 128-142.

Nilsback, M-E. and Zisserman, A. 2006. A Visual Vocabulary for Flower Classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition

Pajares, G. y de la Cruz, J.M. (Eds.) (2010). Aprendizaje Automático: un enfoque práctico. RA-MA, Madrid.

Sheng, X; Tao, F.; Deren L.; Shiwei W. (2010) Object Classification of Aerial Images with Bag-of-Visual Words. *Geoscience and Remote Sensing Letters, IEEE*, 7(2), 266–370

Szelinski, R. (2011) *Computer Vision: Algorithms and Applications*, 1ª ed.; Springer Verlag: Nueva York, USA, pp. 658–729.

Sivic, J. and Zisserman A. (2003) Video Google: A text retrieval approach to object matching in videos. In Proceedings of the International Conference in Computer Vision (ICCV), volume 2, pages 1470–1477.

Vapnik, V. (1995) *The Nature of Statistical Learning Theory*, 2ª ed.; Springer Verlag: Nueva York, USA, pp. 123–170.

Vedaldi, A and Fulkerson, B. (2008). "VLFeat: An Open and Portable Library of Computer Vision Algorithms". Disponible on-line: http://www.vlfeat.org/